Serial Monitor

The Serial Monitor is a tool in the Arduino IDE. It is a tool used to view data sent by the Arduino. It allows communication with the Arduino board via a USB cable. **Through the Serial Monitor, we can read real-time data, see data responses, and send data to the Arduino. This is very useful for debugging and monitoring Arduino projects.**

Components of the Serial Monitor Interface:

- Baud Rate: e.g., 9600 baud
- Input Field: You can enter data to send to the Arduino here. After entering the data, press the enter key or click the "Send" button to transmit it.
- Display Area: This shows the data sent from the Arduino board to the Serial Monitor.

Baud Rate:

The number of symbols transmitted per second in serial communication, usually expressed in bits per second (bps). **It defines the speed of data transmission and determines the frequency at which data is sent and received.** Common baud rates include 9600, 19200, 57600, etc. **Its importance lies in defining the communication speed and efficiency between two devices. In Arduino projects, the baud rate must be consistent to ensure correct data transmission.**

If the baud rates of the sender and receiver are different, data transmission errors will occur, possibly resulting in garbled or lost data. Choosing an appropriate baud rate is important**. For large data transmission, a higher baud rate improves efficiency; lower baud rates are generally more stable and suitable for long-distance communication or environments with**

a lot of interference. Some hardware or modules may only support a specific range of baud rates, which can be found in their usage instructions.

In C++ code, Arduino code is essentially C++. When you see `Serial.begin(9600)` in Arduino code, the part starting with `Serial` is related to the Serial Monitor. `Serial.begin(9600)` initializes serial communication and sets the baud rate to 9600. `Println` sends information to the Serial Monitor.

```
void setup() {
  Serial.begin(9600); // Initialize serial communication and set baud rate to
9600
}

void loop() {
  Serial.print("Hello World"); // Print "Hello World"
  delay(1000); // Delay for 1 second
}
```

Available Serial Functions

- Serial.begin(baudrate): Initializes serial communication and sets the baud rate.
- Serial.end(): Ends serial communication.
- Serial.available(): Returns the number of bytes available for reading.
- Serial.read(): Reads incoming serial data (returns an integer).
- Serial.readString(): Reads the string from the serial buffer until a newline character is encountered.
- Serial.readBytes(): Reads multiple bytes of data from the serial buffer. You can specify the number of bytes to read and store the data in a buffer.
- Serial.flush(): Waits for the transmission of outgoing serial data to complete.
- if (Serial): Checks if the specified serial port is ready.
- Serial.print(value): Sends data without a newline character.
- Serial.println(value): Sends data with a newline character at the end.
- Serial.write(value): Sends binary data. This can be a single byte or a byte array. (The value can be various data types, such as strings, integers, floating-point numbers, etc.)

Connecting the Sensor:

DHT11 temperature sensor, Arduino board, 10kΩ resistor, wires

DHT11 Sensor (3 pins):

- VCC: Power pin (connect to 5V)
- DATA: Data pin (connect to an Arduino digital pin)
- GND: Ground pin

Hardware Connections:

- Connect the VCC pin of the DHT11 sensor to the 5V pin on the Arduino.
- Connect the GND pin of the DHT11 sensor to the GND pin on the Arduino.
- Connect the DATA pin of the DHT11 sensor to a digital pin on the Arduino (D2).
- Connect a 10kΩ pull-up resistor between the VCC and DATA pins of the DHT11.
- Install the DHT library in the Arduino IDE.

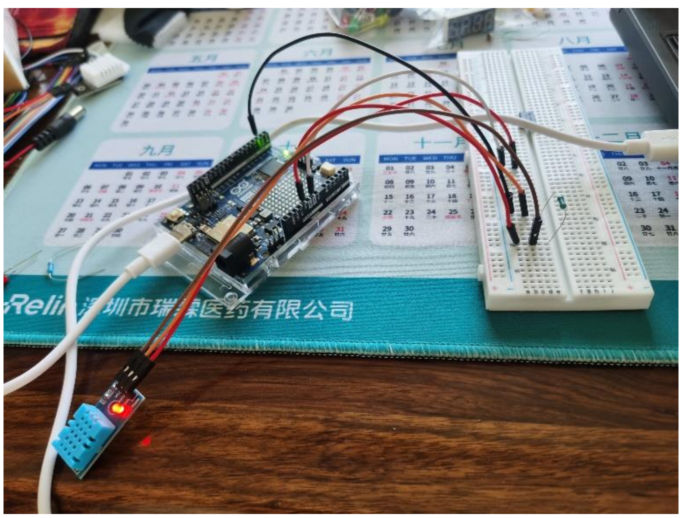
```
#include <DHT.h>

#define DHTPIN 2
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600); // Initialize serial communication at 9600 baud rate
  dht.begin(); // Start the DHT sensor
}

void loop() {
  delay(2000); // Wait for 2 seconds
  float humidity = dht.readHumidity(); // Read humidity
  float temperatureC = dht.readTemperature(); // Read temperature in Celsius
  float temperatureF = dht.readTemperature(true); // Read temperature in
Fahrenheit

  if (isnan(humidity) || isnan(temperatureC) || isnan(temperatureF)) {
    Serial.println("Failed to read from DHT sensor!"); // Print error if reading
fails
    return;
  }

  Serial.print("Humidity: ");
  Serial.print(humidity);
  Serial.print(" %\t");
  Serial.print("Temperature in Celsius: ");
  Serial.print(temperatureC);
```

```
  Serial.print(" *C\t");
  Serial.print("Temperature in Fahrenheit: ");
  Serial.print(temperatureF);
  Serial.println(" *F");
}
```

First, we include the necessary library, the DHT library. Next, we define the pin number and specify that we are using the DHT11 model. **In the setup function, we perform two tasks: initialize serial communication at a baud rate of 9600 and start the DHT sensor.**

In the loop function, we define a float variable to represent non-integer temperature or humidity values. By calling the appropriate functions, we can obtain precise temperature and humidity readings.

We then perform a check using the isnan() function to determine if the sensor is returning valid data. If the sensor fails to read correctly, it will return a null value, which we check with isnan(). If the value is null, we print an error message using Serial.println(). If everything is fine, we proceed to print the results to the Serial Monitor.



```
Output    Serial Monitor ×
Message (Enter to send message to 'Arduino UNO R4 WiFi' on 'COM8')
Humidity: 60.00 %    Temperature in Celsius: 32.60 *C    Temperature in Fahrenheit: 90.68 *F
Humidity: 60.00 %    Temperature in Celsius: 32.40 *C    Temperature in Fahrenheit: 90.32 *F
Humidity: 60.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
Humidity: 60.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
Humidity: 59.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
Humidity: 60.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
Humidity: 60.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
Humidity: 60.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
Humidity: 60.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
Humidity: 60.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
Humidity: 60.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
Humidity: 60.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
Humidity: 60.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
Humidity: 60.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
Humidity: 60.00 %    Temperature in Celsius: 32.30 *C    Temperature in Fahrenheit: 90.14 *F
```